call void @llvm.dbg.value(metadata ptr %a, metadata !49, metadata ...!DIExpression()),!dbg!58 call void @llvm.dbg.value(metadata ptr %b, metadata !50, metadata ...!DIExpression()),!dbg!58 call void @llvm.dbg.value(metadata ptr %c, metadata !51, metadata ...!DIExpression()),!dbg!58 call void @llvm.dbg.value(metadata ptr %cond, metadata !52, metadata ...!DIExpression()),!dbg!58 call void @llvm.dbg.value(metadata i32 %n, metadata !53, metadata ... !DIExpression()), !dbg !58 %call = tail call double @getTimeMiliSeconds(), !dbg !59 call void @llvm.dbg.value(metadata double %call, metadata !55, metadata ...!DIExpression()),!dbg!58 %0 = load i32, ptr @EventSet, align 4, !dbg !60, !tbaa !62 %call1 = tail call i32 @PAPI start(i32 noundef %0) #13, !dbg !66 call void @llvm.dbg.value(metadata i32 %call1, metadata !54, metadata ... !DIExpression()), !dbg !58 %cmp.not = icmp eq i32 %call1, 0, !dbg !67 br i1 %cmp.not, label %for.cond.preheader, label %if.then, !dbg !68 if.then: for.cond.preheader: %4 = load ptr, ptr @stderr, align 8, !dbg !73, !tbaa !75 call void @llvm.dbg.value(metadata i32 0, metadata !56, metadata %call2 = tail call i32 (ptr, ptr, ...) @fprintf(ptr noundef %4, ptr noundef ... nonnull @.str, i32 noundef %call1, ptr noundef nonnull @.str.1, i32 noundef ..!DIExpression()),!dbq!69 %cmp3131 = icmp sgt i32 %n, 0, !dbg !70 ... 37) #14, !dbg !73 br i1 %cmp3131, label %for.body.preheader, label %for.cond.cleanup, !dbg !72 tail call void @exit(i32 noundef %call1) #15, !dbg !73 F unreachable, !dbg !73 for.body.preheader: %wide.trip.count = zext i32 %n to i64, !dbg !70 %1 = tail call i32 @llvm.vscale.i32(), !dbg !72 %2 = shl i32 %1, 2, !dbg !72 %3 = shl i32 %1, 3, !dbg !72 %.not = icmp ugt i32 %3, %n, !dbg !72 br i1 %.not, label %for.body.preheader27, label %pre.alc, !dbg !72 $\sqrt{14}$ = tail call <vscale x 4 x i1> @llvm.aarch64.sve.ptrue.nxv4i1(i32 31) %15 = tail call <vscale x 4 x i32> @llvm.aarch64.sve.index.nxv4i32(i32 0, ... i32 1) %16 = urem i32 %n, %2 %total.iterations.to.be.vectorized = sub nsw i32 %n, %16 %17 = load < vscale x 4 x i8 >, ptr %cond, align 4 %18 = icmp ne < vscale x 4 x i8 > %17, zeroinitializer%19 = tail call i64 @llvm.aarch64.sve.cntp.nxv4i1(<vscale x 4 x i1> %18, .. <vscale x 4 x i1> %18) %20 = trunc i64 %19 to i32 br label %alc.header alc.header: %vector.loop.index = phi i32 [%2, %pre.alc], [%100, %new.latch] %uniform.vector = phi <vscale x 4 x i32> [%15, %pre.alc], [%97, .. %new.latch] %uniform.vector.predicates = phi <vscale x 4 x i1> [%18, %pre.alc], [%98, .. %new.latch] %uniform.vec.actives = phi i32 [%20, %pre.alc], [%99, %new.latch] %21 = tail call <vscale x 4 x i32> @llvm.aarch64.sve.index.nxv4i32(i32 ... %vector.loop.index, i32 1) %22 = sext i32 %vector.loop.index to i64, !dbg !82 %23 = getelementptr inbounds i8, ptr %cond, i64 %22, !dbg !82 %24 = load < vscale x 4 x i8 > , ptr %23, align 4 %25 = icmp ne < vscale x 4 x i8 > %24, zeroinitializer%26 = tail call i64 @llvm.aarch64.sve.cntp.nxv4i1(<vscale x 4 x i1> %25, ... <vscale x 4 x i1> %25) %27 = trunc i64 %26 to i32 %28 = add i32 %uniform.vec.actives, %27 %condition.not = icmp ugt i32 %28, %2 br i1 %condition.not, label %linearized, label %lane.gather lane.gather: %29 = tail call <vscale x 4 x i32> @llvm.aarch64.sve.compact.nxv4i32(<vscale ... x 4 x i1> %uniform.vector.predicates, <vscale x 4 x i32> %uniform.vector) %30 = tail call <vscale x 4 x i32> @llvm.aarch64.sve.compact.nxv4i32(<vscale ... x 4 x i1> %25, <vscale x 4 x i32> %21) %31 = tail call <vscale x 4 x i1> @llvm.aarch64.sve.whilelt.nxv4i1.i32(i32 ... 0, i32 %uniform.vec.actives) %32 = tail call <vscale x 4 x i32> @llvm.aarch64.sve.splice.nxv4i32(<vscale ... x 4 x i1> %31, <vscale x 4 x i32> %29, <vscale x 4 x i32> %30) %33 = tail call <vscale x 4 x i1> @llvm.aarch64.sve.whilelt.nxv4i1.i32(i32 $\%34 = tail\ call\ i64\ @llvm.aarch64.sve.cntp.nxv4i1(< vscale x 4 x i1 > \%33,$ \dots <vscale x 4 x i1> %33) %35 = trunc i64 %34 to i32 %36 = icmp ult i32 %28, %2 br i1 %36, label %new.latch, label %uniform.block uniform.block: %64 = tail call < vscale x 4 x i32 >... @llvm.aarch64.sve.ld1.gather.sxtw.index.nxv4i32(<vscale x 4 x i1> %14, ptr ... %c, <vscale x 4 x i32> %32) %65 = tail call < vscale x 4 x i32 >linearized: ... @llvm.aarch64.sve.ld1.gather.sxtw.index.nxv4i32(<vscale x 4 x i1> %14, ptr %37 = getelementptr inbounds i32, ptr %a, i64 %22, !dbg !89 ... %b, <vscale x 4 x i32> %32) %38 = getelementptr inbounds i32, ptr %c, i64 %22, !dbg !91 %66 = mul < vscale x 4 x i32 > %65, %32%39 = getelementptr inbounds i32, ptr %b, i64 %22, !dbg !92 %40 = tail call <vscale x 4 x i32> @llvm.masked.load.nxv4i32.p0(ptr %38, i32 %67 = mul <vscale x 4 x i32> %64, shufflevector (<vscale x 4 x i32> ... insertelement (<vscale x 4 x i32> undef, i32 -2, i64 0), <vscale x 4 x i32> ... 16, <vscale x 4 x i1> %25, <vscale x 4 x i32> undef) .. undef, <vscale x 4 x i32> zeroinitializer) $%41 = tail\ call\ < vscale\ x\ 4\ x\ i32 > @llvm.masked.load.nxv4i32.p0(ptr\ %39, i32)$ %68 = shl < vscale x 4 x i32 > %32, shufflevector (< vscale x 4 x i32 > %32) ... 16, <vscale x 4 x i1> %25, <vscale x 4 x i32> undef) .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> %42 = mul < vscale x 4 x i32 > %41, %21.. undef, <vscale x 4 x i32> zeroinitializer) %43 = mul < vscale x 4 x i 32 > %40, shufflevector (< vscale x 4 x i 32 > %40) %69 = add < vscale x 4 x i32 > %65, %68... insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 -2, i64 0), $\langle vscale \times 4 \times i32 \rangle$ %70 = add < vscale x 4 x i32 > %69, %67... undef, <vscale x 4 x i32> zeroinitializer) %71 = add < vscale x 4 x i32 > %70, %66%44 = shl < vscale x 4 x i32 > %21, shufflevector (< vscale x 4 x i32 > %21) tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 ... insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 1, i64 0), $\langle vscale \times 4 \times i32 \rangle$... x i32> %71, <vscale x 4 x i1> %14, ptr %a, <vscale x 4 x i32> %32) ... undef, <vscale x 4 x i32> zeroinitializer) %72 = sub <vscale x 4 x i32> shufflevector (<vscale x 4 x i32> insertelement %45 = add < vscale x 4 x i32 > %41, %44.. (<vscale x 4 x i32> undef, i32 -3, i64 0), <vscale x 4 x i32> undef, <vscale %46 = add <vscale x 4 x i32> %45, %43 .. x 4 x i32> zeroinitializer), %64 %47 = add < vscale x 4 x i32 > %46, %42%73 = mul < vscale x 4 x i32 > %72, %32tail call void @llvm.masked.store.nxv4i32.p0(<vscale x 4 x i32> %47, ptr %74 = add < vscale x 4 x i32 > %65, %64.. %37, i32 16, <vscale x 4 x i1> %25) %75 = sub < vscale x 4 x i32 > %71, %74%48 = sub <vscale x 4 x i32> shufflevector (<vscale x 4 x i32> insertelement %76 = shl < vscale x 4 x i32 > %75, shufflevector (< vscale x 4 x i32 > %75) ... (<vscale x 4 x i32> undef, i32 -3, i64 0), <vscale x 4 x i32> undef, <vscale .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> ... $\times 4 \times i32 > zeroinitializer)$, %40 .. undef, <vscale x 4 x i32> zeroinitializer) %49 = mul < vscale x 4 x i32 > %48, %21%77 = add < vscale x 4 x i32 > %73, shufflevector (< vscale x 4 x i32 > %73) %50 = add < vscale x 4 x i32 > %41, %40... insertelement (<vscale x 4 x i32> undef, i32 2, i64 0), <vscale x 4 x i32> %51 = sub < vscale x 4 x i32 > %47, %50.. undef, <vscale x 4 x i32> zeroinitializer) %52 = shl < vscale x 4 x i 32 > %51, shufflevector (< vscale x 4 x i 32 > %51) %78 = add < vscale x 4 x i32 > %77, %76... insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 1, i64 0), $\langle vscale \times 4 \times i32 \rangle$ tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 ... undef, <vscale x 4 x i32> zeroinitializer) ... x i32> %78, <vscale x 4 x i1> %14, ptr %b, <vscale x 4 x i32> %32) %53 = add < vscale x 4 x i32 > %49, shufflevector (< vscale x 4 x i32 > %49) %79 = sub < vscale x 4 x i32 > %64, %78... insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 2, i64 0), $\langle vscale \times 4 \times i32 \rangle$ %80 = shl < vscale x 4 x i32 > %79, shufflevector (< vscale x 4 x i32 > %79) ... undef, <vscale x 4 x i32> zeroinitializer) .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> %54 = add <vscale x 4 x i32> %53, %52 ... undef, <vscale x 4 x i32> zeroinitializer) tail call void @llvm.masked.store.nxv4i32.p0(<vscale x 4 x i32> %54, ptr %81 = mul < vscale x 4 x i32 > %32, %32... %39, i32 16, <vscale x 4 x i1> %25) %82 = add < vscale x 4 x i32 > %80, %81%55 = sub < vscale x 4 x i32 > %40, %54%83 = mul < vscale x 4 x i32 > %82, shufflevector (< vscale x 4 x i32 > %82) %56 = shl < vscale x 4 x i32 > %55, shufflevector (< vscale x 4 x i32 > %55) .. insertelement (<vscale x 4 x i32> undef, i32 -3, i64 0), <vscale x 4 x i32> ... insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) ... undef, <vscale x 4 x i32> zeroinitializer) %84 = sub <vscale x 4 x i32> %71, %32 %57 = mul < vscale x 4 x i32 > %21, %21%85 = add <vscale x 4 x i32> %84, %78 %58 = add < vscale x 4 x i32 > %56, %57%86 = shl < vscale x 4 x i32 > %85, shufflevector (< vscale x 4 x i32 >%59 = mul <vscale x 4 x i32> %58, shufflevector (<vscale x 4 x i32> .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> ... insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 -3, i64 0), $\langle vscale \times 4 \times i32 \rangle$.. undef, <vscale x 4 x i32> zeroinitializer) ... undef, <vscale x 4 x i32> zeroinitializer) %87 = add < vscale x 4 x i32 > %83, %86%60 = sub < vscale x 4 x i32 > %47, %21tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 %61 = add < vscale x 4 x i32 > %60, %54... x i32> %87, <vscale x 4 x i1> %14, ptr %c, <vscale x 4 x i32> %32) %62 = shl < vscale x 4 x i32 > %61, shufflevector (< vscale x 4 x i32 > %61) %88 = add i32 %vector.loop.index, %2 ... insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> %89 = tail call < vscale x 4 x i32 > @llvm.aarch64.sve.index.nxv4i32(i32 %88,... undef, <vscale x 4 x i32> zeroinitializer) .. i32 1) %63 = add < vscale x 4 x i32 > %59, %62%90 = sext i 32 % 88 to i 64, !dbg !82tail call void @llvm.masked.store.nxv4i32.p0(<vscale x 4 x i32> %63, ptr %91 = getelementptr inbounds i8, ptr %cond, i64 %90, !dbg !82 ... %38, i32 16, <vscale x 4 x i1> %25) %92 = load <vscale x 4 x i8>, ptr %91, align 4 br label %new.latch %93 = icmp ne < vscale x 4 x i8 > %92, zeroinitializer%94 = tail call i64 @llvm.aarch64.sve.cntp.nxv4i1(<vscale x 4 x i1> %93, .. <vscale x 4 x i1> %93) %95 = trunc i64 %94 to i32 br label %new.latch new.latch: %96 = phi i32 [%vector.loop.index, %linearized], [%vector.loop.index, ... %lane.gather], [%88, %uniform.block] %97 = phi < vscale x 4 x i32 = [%uniform.vector, %linearized], [%32,... %lane.gather], [%89, %uniform.block] %98 = phi <vscale x 4 x i1> [%uniform.vector.predicates, %linearized], [.. %33, %lane.gather], [%93, %uniform.block] %99 = phi i32 [%uniform.vec.actives, %linearized], [%35, %lane.gather], ... [%95, %uniform.block] %100 = add i32 %96, %2%.not3 = icmp ult i32 %100, %total.iterations.to.be.vectorized br i1 %.not3, label %alc.header, label %middel.block middel.block: %condition1 = icmp eq i32 %16, 0 %101 = tail call < vscale x 4 x i32 >... @llvm.aarch64.sve.ld1.gather.sxtw.index.nxv4i32(<vscale x 4 x i1> %98, ptr .. %c, <vscale x 4 x i32> %97) %102 = tail call < vscale x 4 x i32 >... @llvm.aarch64.sve.ld1.gather.sxtw.index.nxv4i32(<vscale x 4 x i1> %98, ptr ... %b, <vscale x 4 x i32> %97) %103 = mul < vscale x 4 x i32 > %102, %97%104 = mul < vscale x 4 x i32 > %101, shufflevector (< vscale x 4 x i32 > %101) .. insertelement (<vscale x 4 x i32> undef, i32 -2, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) %105 = shl < vscale x 4 x i32 > %97, shufflevector (< vscale x 4 x i32 > %97) .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) %106 = add < vscale x 4 x i32 > %102, %105%107 = add < vscale x 4 x i32 > %106, %104%108 = add < vscale x 4 x i32 > %107, %103tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 ... x i32> %108, <vscale x 4 x i1> %98, ptr %a, <vscale x 4 x i32> %97) %109 = sub <vscale x 4 x i32> shufflevector (<vscale x 4 x i32> .. insertelement (<vscale x 4 x i32> undef, i32 -3, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer), %101 %110 = mul < vscale x 4 x i32 > %109, %97 %111 = add < vscale x 4 x i32 > %102, %101%112 = sub < vscale x 4 x i32> %108, %111 %113 = shl < vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4 x i32 > %112, shufflevector (<math>< vscale x 4.. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) %114 = add <vscale x 4 x i32> %110, shufflevector (<vscale x 4 x i32> .. insertelement (<vscale x 4 x i32> undef, i32 2, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) %115 = add <vscale x 4 x i32> %114, %113 tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 .. x i32> %115, <vscale x 4 x i1> %98, ptr %b, <vscale x 4 x i32> %97) %116 = sub <vscale x 4 x i32> %101, %115 %117 = shl < vscale x 4 x i 32 > %116, shufflevector (< vscale x 4 x i 32 > %116) .. insertelement (<vscale x 4 x i32> undef, i32 1, i64 0), <vscale x 4 x i32> .. undef, <vscale x 4 x i32> zeroinitializer) %118 = mul < vscale x 4 x i32 > %97, %97%119 = add < vscale x 4 x i32 > %117, %118%120 = mul < vscale x 4 x i 32 > %119, shufflevector (< vscale x 4 x i 32 > %12 > %12 = %.. insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 -3, i64 0), $\langle vscale \times 4 \times i32 \rangle$.. undef, <vscale x 4 x i32> zeroinitializer) %121 = sub <vscale x 4 x i32> %108, %97 %122 = add <vscale x 4 x i32> %121, %115 %123 = shl < vscale x 4 x i 32 > %122, shufflevector (< vscale x 4 x i 32 > %123) .. insertelement ($\langle vscale \times 4 \times i32 \rangle$ undef, i32 1, i64 0), $\langle vscale \times 4 \times i32 \rangle$.. undef, <vscale x 4 x i32> zeroinitializer) %124 = add < vscale x 4 x i32 > %120, %123tail call void @llvm.aarch64.sve.st1.scatter.sxtw.index.nxv4i32(<vscale x 4 .. x i32> %124, <vscale x 4 x i1> %98, ptr %c, <vscale x 4 x i32> %97) %125 = zext i32 %100 to i64br i1 %condition1, label %for.cond.cleanup, label %for.body.preheader27 for.body.preheader27: %indvars.iv.ph = phi i64 [%125, %middel.block], [0, %for.body.preheader] br label %for.body, !dbg !72 for.body: %indvars.iv = phi i64 [%indvars.iv.next, %for.inc], [%indvars.iv.ph, ... %for.body.preheader27] call void @llvm.dbg.value(metadata i64 %indvars.iv, metadata !56, metadata ...!DIExpression()), !dbg !69 %arrayidx = getelementptr inbounds i8, ptr %cond, i64 %indvars.iv, !dbg !82 %6 = load i8, ptr %arrayidx, align 1, !dbg !82, !tbaa !85, !range !87 %tobool.not = icmp eq i8 %6, 0, !dbg !82 br i1 %tobool.not, label %for.inc, label %if.then4, !dbg !88 if.then4: %arrayidx6 = getelementptr inbounds i32, ptr %a, i64 %indvars.iv, !dbg !89 %arrayidx8 = getelementptr inbounds i32, ptr %c, i64 %indvars.iv, !dbg !91 %7 = load i32, ptr %arrayidx8, align 4, !dbg !91, !tbaa !62 %arrayidx11 = getelementptr inbounds i32, ptr %b, i64 %indvars.iv, !dbg !92 %8 = load i32, ptr %arrayidx11, align 4, !dbg !92, !tbaa !62 %9 = trunc i64 %indvars.iv to i32, !dbg !93 %mul21 = mul nsw i32 %8, %9, !dbg !93 %reass.add = sub i32 %9, %7 %reass.mul = shl i32 %reass.add, 1 %add = add i32 %mul21, %8, !dbg !94 %add25 = add i32 %add, %reass.mul, !dbg !95 store i32 %add25, ptr %arrayidx6, align 4, !dbg !95, !tbaa !62 %mul43127.neg = sub i32 -3, %7, !dbg !96 %add44.neg134 = mul i32 %mul43127.neg, %9, !dbg !96 %10 = add i32 %8, %7 %reass.add129 = sub i32 %add25, %10 %reass.mul130 = shl i32 %reass.add129, 1 %add37 = add i32 %add44.neg134, 2, !dbg !97 %sub47 = add i32 %add37, %reass.mul130, !dbg !98 store i32 %sub47, ptr %arrayidx11, align 4, !dbg !98, !tbaa !62 %11 = sub nsw i32 %7, %sub47, !dbg !99 %sub61 = shl nsw i32 %11, 1, !dbg !99 %mul62 = mul nsw i32 %9, %9, !dbg !100 %add63 = add nsw i32 %sub61, %mul62, !dbg !101 %mul64.neg = mul i32 %add63, -3, !dbg !102 %12 = sub i32 %add25, %9, !dbg !103 %13 = add i32 %12, %sub47, !dbg !104 %sub65 = shl i32 %13, 1, !dbg !104 %sub71 = add i32 %mul64.neg, %sub65, !dbg !105 store i32 %sub71, ptr %arrayidx8, align 4, !dbg !105, !tbaa !62 br label %for.inc, !dbg !106 for.inc: %indvars.iv.next = add nuw nsw i64 %indvars.iv, 1, !dbg !107 call void @llvm.dbg.value(metadata i64 %indvars.iv.next, metadata !56, ... metadata !DIExpression()), !dbg !69 %exitcond.not = icmp eq i64 %indvars.iv.next, %wide.trip.count, !dbg !70 br i1 %exitcond.not, label %for.cond.cleanup, label %for.body, !dbg!72, ... !llvm.loop !108 for.cond.cleanup: %5 = load i32, ptr @EventSet, align 4, !dbg !77, !tbaa !62 %call73 = tail call i32 @PAPI_stop(i32 noundef %5, ptr noundef nonnull ... @CounterValues) #13, !dbg !79 call void @llvm.dbg.value(metadata i32 %call73, metadata !54, metadata ...!DIExpression()),!dbg!58 %cmp74.not = icmp eq i32 %call73, 0, !dbg !80 br i1 %cmp74.not, label %if.end77, label %if.then75, !dbg !81 F if.then75: if.end77: %126 = load ptr, ptr @stderr, align 8, !dbg !112, !tbaa !75 %call76 = tail call i32 (ptr, ptr, ...) @fprintf(ptr noundef %126, ptr %call78 = tail call double @getTimeMiliSeconds(), !dbg !114 ... noundef nonnull @.str, i32 noundef %call73, ptr noundef nonnull @.str.1, i32 %sub79 = fsub double %call78, %call, !dbg !115 ... noundef 53) #14, !dbg !112 store double %sub79, ptr @ExecutionTime, align 8, !dbg !116, !tbaa !117 tail call void @exit(i32 noundef %call73) #15, !dbg !112 ret void, !dbg !119 unreachable, !dbg !112